Proceedings of the 24th UK Conference of the
Association for Computational Mechanics in Engineering
31 March - 01 April 2016, Cardiff University, Cardiff

A Contact Detection Code using Triangles for Non-Spherical Particle Simulations

***Konstantinos Krestenitis[1], Tobias Weinzierl[1] and Tomasz Koziara[1]**

[1]School of Engineering and Computing Sciences, University of Durham, DH1 3LE, Durham

*konstantinos.krestenitis@durham.ac.uk

ABSTRACT

We present a novel DEM (discrete element method) code with explicit time stepping. DEM codes simulate billions of small particles that interact with each other primarily through collisions. Different to state-of-the-art codes, we rely on triangulated non-spherical particles. This is computationally demanding, and we thus devise an asynchronous data exchange communication technique using MPI (Message Passing Interface) on manycore supercomputers, we discuss possible solutions to handle ghost particles that overlap multiple subdomains. At the compute node level, shared memory parallelism as well as vectorised SIMD executions are studied. On the algorithmic side, we explore a hybrid parallelisation approach and memory layouts that are suited to combine robust exact geometry checks with a fast penalty-based method.

Key Words: *SIMD; Contact Detection; MPI; Discrete Element Method*

1. Introduction

In contact mechanics, fluid-stucture interaction or other fields, it is an essential task to compute the distance between geometries to determine contact. We present a Discrete Element Method (DEM) code that simulates particles that interact through spring contact. DEM for example is used to study granular particles in environmental or medical engineering. Different to state-of-the-art codes, we do not use spheres to model the geometry [2] but rely on triangles to model meshed surfaces. Non-spherical particles promise to facilitate more accurate physics than sphere-based approaches [5]. The focus on triangles for rigid body contact dynamics rather than arbitrary polygons simplifies geometric checks and facilitates memory layouts allowing vectorised computation [5]. It is vital to obtain performance from current and upcoming architectures to enable engineers to simulate more realistic materials.

To achieve this we propose a hybrid method that combines the advantages of two triangle-to-triangle distance calculation methods. The method benefits from the performance of an iterative Newton-penalty convergence solver and the robustness of an all-to-all brute force geometric primitive comparison method [5]. We then make this method exploit multi-core and SIMD (Single Instruction Multiple Data) environments [4]. Finally we use the Message Passing Interface (MPI) and Recursive Coordinate Bisection (RCB) load balancing [1] to migrate particles over the HPC cluster nodes while we execute in parallel the contact detection stage of the simulation. The spatial domain decomposition and migration/movement of data requires a careful analysis of data alignment and data dependencies. We use asynchronous non-blocking communication such that there are no waiting delays during the data exchange.

2. Hybrid Vectorised Parallel Contact Detection

Contact detection for non-spherical particles requires us to find the distances between the triangles. Our implementation combines two methods, the brute force method and the iterative Newton-Raphson based on a penalty method for the inequality constraints. The brute force method computes the minimum distance between vertex-to-triangle, and segment-to-segment geometric combinations. Brute force always finds the correct solution but lacks performance as it involves many case distinctions. The penalty method parameterises the distance problem as $f\,(a,b,c,d)_{min} = \|x\,(a,b) - y\,(c,d)\|^2$ where x and y are the barycentric functions of two triangles and it creates a minimization problem to solve. The penalty method

also introduces the penalty parameter r for the steepness of the merit function outside from the feasible domain. In (1) c is the constraint function with six constraints.

$$P(x) = f(x) + r \sum_{i=1\ldots6} max(0, c(x_i))^2 \tag{1}$$

We propose a new hybrid algorithmic approach that combines iterative speed and the robustness of a brute force method. The hybrid 'Newton-Brute Force' solver first runs a penalty solver. If convergence is not achieved within a specified number of iterations (typically four) it falls back to the robust brute force. The decision to fall back is made when the difference between two Newton steps is not within a specified tolerance level.

We split the computational work into batches of fixed size. A batch is a set of triangle pairs that have to be checked against each other. Every batch passes through the penalty solver with SIMD. The handling of multiple triangle comparisons thus can be fused, which allows us to exploit wide vector registers. Convergence checks and fall-back to brute force is done on a per-batch basis. Using multiple cores/threads, it is possible to execute the batches in parallel.

Algorithm 1 DEM Simulation Pseudo code

1. Load balance triangles //assigment to subdomains
2. Migrate triangles to MPI network using blocking communication //balancing movement
3. Search overlapping ghost triangles to send
4. Initiate neighbors asynchronous MPI send/receive
5. Local all to all triangle contact detection
6. Retrieve required ghost triangles from neighbors
7. Local all to ghost triangle contact detection //data exchange
8. Wait for neighborhood asynchronous communication to terminate (No Real Wait)
9. Derive contact forces from contact points generated from contact detection
11. Explicit time integration

For many-node supercomputers we decompose the computational domain into subdomains. Alternative decomposition approaches in the literature such as force decomposition that are applied in molecular dynamics (MD) and smoothed particle hydrodynamics (SPH) simulations [2, 3] are not benchmarked. Our decomposition is based on the spatial position of triangle points using Recursive Coordinate Bisection (RCB) [1]. Each triangle thus is owned - and persistently stored-exclusively on one rank. Copies are exchanged per time step where triangles overlap into neighboring domains.

Algorithm 1 describes the whole DEM simulation and how MPI communication is realised. Data transfers between subdomain ranks on MPI use blocking synchronous and asynchronous non-blocking communication at different stages of the DEM simulation. During the data migration/movement stage blocking synchronous communication is used to move local triangles to neighboring processes to realise load re-balancing. Asynchronous communication is used to exchange ghost triangles between neighbors to maintain streamlined computation while minimizing communication waiting overhead.

3. Implementation

It is uncertain whether the penalty iterations converge for every batch in four iterations. This creates a couple of implementation challenges that affect performance. There are load balancing implications with OpenMP due to the non-deterministic nature of the algorithm. It is not possible to know a priori the balancing of such a non-deterministic algorithm. The uncertain runtime cost stems from the geometry of the problem. A solution is to rely on dynamic scheduling. Investigations however show that dynamic balancing does not show signs of performance improvement and this is subject to further investigation. It remains an open question whether static scheduling is sufficient.

Another tuning parameter is the batch/grain size. The grain size behavior affects the tolerance value and the SIMD work performed by both penalty and brute force methods. The bigger the more triangle

Algorithm 2 Hybrid Triangle Distance Computation Algorithm

```
1. FOR batches of grain size X loop through shared memory threads:
2.     run penalty-based Newton penalty solver
3.     IF batchError [i] > 1E-8 //fixed error value
4.       FOR batches of grain size X loop through shared memory threads:
5.         run brute force segment to segment, point to triangle comparison solver
6.       ENDFOR
7.     ENDIF
8. ENDFOR
```

comparisons can be fused into SIMD statements. However, if only one triangle fails to converge, the whole batch falls back to brute force. Tuning the grain size is nontrivial. Trial and error fine-tuning over a given set of random triangles worked best for static load balancing. The batch size for random triangles is optimal at size eight.

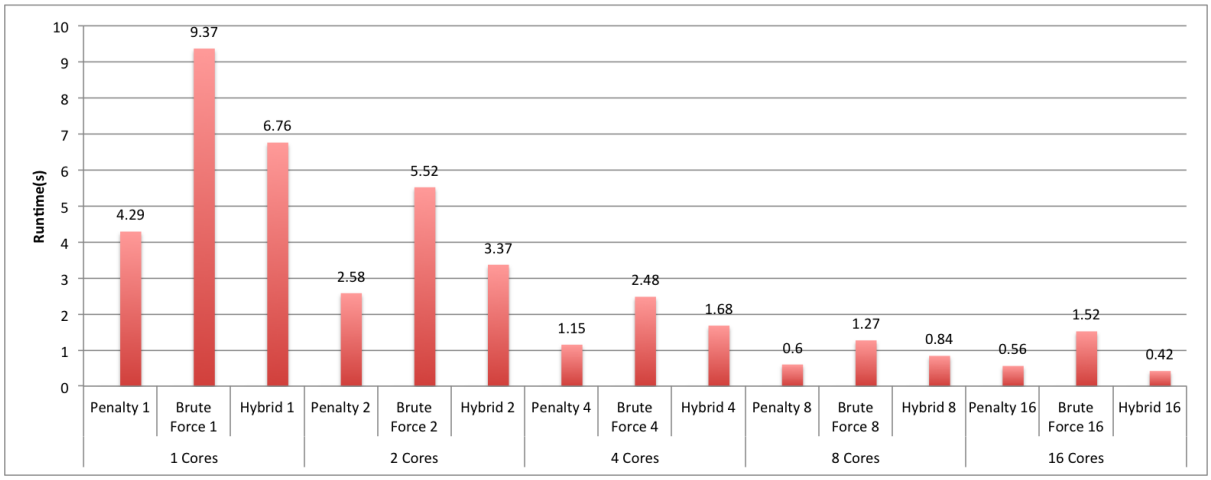

Figure 1: Run-time comparison of the penalty, brute force and the hybrid method over set of cores (OpenMP 4.0).

The hybrid approach performance is shown in Figure 1 where multi-core executions run on all cores. Its performance is settled between brute force and penalty. The hybrid method is slower than penalty because there are batches of triangle pairs that do not reach convergence tolerance. It is faster than brute force because if the batch size is optimal no significant number of batches fail the tolerance criterion. The hybrid algorithm scales over the second socket of the CPU. It is observed that brute force does not gain significantly between eight and sixteen cores because of bandwidth stagnation at the socket interconnect. Theoretically, performance can be tuned further by investigating the load balancing implications of the non-deterministic element of the algorithm.

A major challenge is sustaining a balanced workload while minimizing inter-node communication and maximizing intra-node vectorisation. Splitting the spatial domain and assigning subdomains to cores introduces data dependencies between neighboring ranks. Subdomains have to retrieve surface triangles that overlap into neighboring subdomains. We rely on the state-of-the-art load balancer Zoltan [1] and focus exclusively on the compute phase of the algorithm.

In Figure 2 we measure the total MPI waiting time of ranks for neighbor data exchange if no non-blocking MPI is used. We see an increase with the number of rank. This motivates our non-blocking scheme where neighbour data exchange is triggered before any local computation starts. Once all local computations terminate, the distance of local triangles against remote ghosts are computed to get the overall local domain contacts. The non-blocking communication strategy scales perfectly, if the local contact detection takes longer than the transmission of the ghost data exchange.

All performance tests in Figures 1 and 2 use the AVX instruction set on an Intel Sandy Bridge 2.0GHz i5

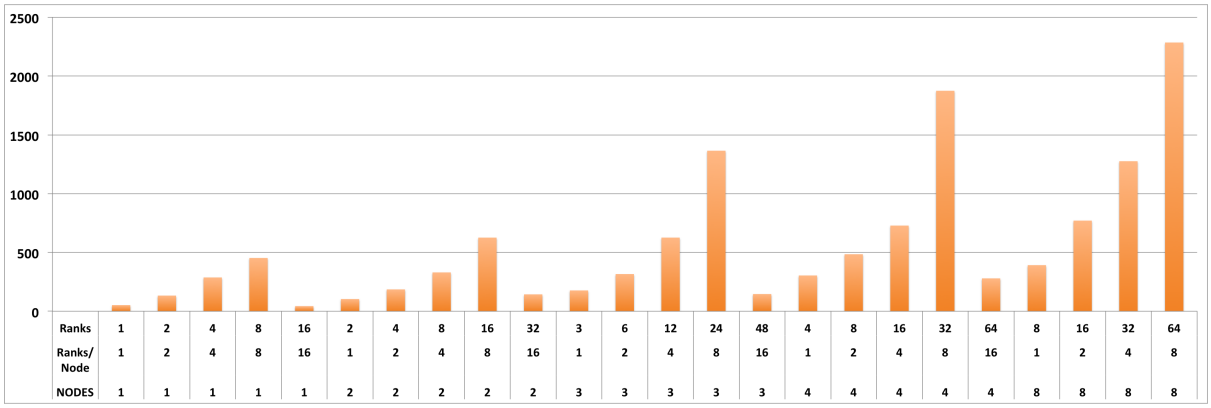

Figure 2: Waiting time [t]=s per MPI rank/node for all-to-all neighbor data exchange over 1000 timesteps (25 mil triangles, 10k non-spherical particles).

CPU with 600 MHz DDR3 RAM. The performance test comprises computing the distance of ten million pairs of multiple lengths.

4. Conclusion

The research provides insight to a hybrid solver that offers robustness and speed for triangle-based DEM algorithms. We study MPI communication patterns and the implications on data layouts using asynchronous non-blocking communication to avoid waiting delays during exchange of ghost triangles. We seem to obtain reasonable scalability and performance for triangle-based particle realisation but many important questions with respect to load balancing remain open.

5. Acknowledgments

This work has been sponsored by EPSRC (Engineering and Physical Sciences Research Council) and EDF Energy as part of an ICASE studentship. This work also made use of the facilities of N8 HPC provided and funded by the N8 consortium and EPSRC (Grant No. N8HPC_DUR_TW_PEANO). The Centre is co-ordinated by the Universities of Leeds and Manchester. We also thank University of Durham for the supercomputing resources and technical assistance. All underlying software is open source and available at [6].

References

[1] Erik G. Boman, Umit V. Catalyurek, Cedric Chevalier, and Karen D. Devine. The Zoltan and Isorropia parallel toolkits for combinatorial scientific computing: Partitioning, ordering and coloring. Sci. Program. 20, 2 (April 2012), 129-150. DOI=http://dx.doi.org/10.1155/2012/713587

[2] S. Plimpton, Fast Parallel Algorithms for Short-Range Molecular Dynamics, J Comp Phys, 117, 1-19 (1995).

[3] Shaw, D.E. A fast, scalable method for the parallel evaluation of distance-limited pairwise particle interactions. Journal of Computational Chemistry, 26(13):1318-1328, 2005.

[4] Shuo Li, Brian Rea, Jim Cownie. Extending Parallelism from Intel Xeon Processor to Intel Xeon Phi Coprocessor: A Structured, Stepwise Approach to Manycore Programming. Intel Corporation. 2014

[5] Krestenitis, K. Koziara, T. Calculating the minimum distance between triangles on SIMD Hardware. ACME. Swansea, United Kingdom. 2015

[6] Krestenitis, K. et al. Delta - A Contact Detection Framework for Non-Spherical Particle DEM Simulations. 2016. http://ikonstantinos.com/delta/.